情緒辨識

組別：第八組

組長：洪翊禎

製作專題報告書之組員：羅文昱、蔡佩穎

無製作專題報告書之組員：劉亦琇、吳學賢、江亮瑩、劉威廷

指導老師：張志勇老師、蒯思齊老師

日期：2024/06/30

班別：半導體 AI 與 ChatGPT 跨領域班 第六梯次

目錄

小組成員介紹

1. 109 洪翊禎
    - 工作崗位：國軍部隊政戰官（待退）
    - 參與課程的動機：轉換跑道
    - 心得分享：大學是念心理學系，課程對我來說真的是非常的跨領域，但會努力學習，希望能將所學結合。
    - 專題分工：將專題簡報匯整至專題報告書。

2. 113 羅文昱
    - 工作崗位：目前任職台灣住重離子科技股份有限公司擔任應用工程師
    - 參與課程的動機：參與課程的動機是希望能透過學習 python 理解機械學習及深度學習，拓展我對資料分析的想法及對工作上有更多不同的應用，甚至期待熟悉後能夠跨出現在的工作領域有不同面向的發展。
    - 心得分享：雖然現在的工作是接觸機台參數與資料分析有關，但與過去在學期間的科系關聯性不高，從這堂課學習 python、機械學習及深度學習並對其運作的原理有了初步的理解，課程中也有利用不同的專題讓我們接觸不同的應用，接下來我會嘗試去運用，並在未來結合工作或是探索其他的領域。
    - 專題分工：**製作專題簡報及專題報告書**。

3. 117 蔡佩穎
    - 工作崗位：陽明交通大學臨床腦科學所碩士生
    - 參與課程的動機：因目前加入實驗室是研究情緒相關的主題，運用 EEG, MRI 等儀器收集資料去做研究，但因為今年即將入學所以報名此課程，希望可以有些基礎，有助於未來學習。
    - 心得分享：其實我大學是生科畢業，對於資訊這方面很陌生，但對於人腦及情緒等方面有研究興趣，因此開始學習 AI，可以了解人腦運作，藉此探討有關情緒等研究，課程中我覺得 machine learning and deep learning 比較難，還在學習當中，我也在反覆思考這對於未來研究該如何應用，且目前也開始查閱有關 emotion paper 等等。我認為上這門課非常重要，是培養基本功，在專題研究也是情緒相關主題，這對我來說意義重大。
    - 專題分工：**製作專題簡報及專題報告書**。

4. 118 劉亦琇
    - 工作崗位：目前在一家載板公司擔任技術師
    - 參與課程的動機：因對現階段的工作不滿意，有轉職的想法，因此想利用空閒時間學習新的技能。之前工作有撰寫簡單的程式，並對此產生了

興趣。經過網路搜尋，並想到 AI 產業正是熱門的領域，最終選擇了這門涵蓋 Python 入門、機器學習、深度學習、資料處理與視覺化，以及 ChatGPT 應用的課程。

- 心得分享：課程內容豐富也很有挑戰性，對於 AI 的技術和前景也有更深度的了解。讓我在職涯上多了可以考慮發展的方向。謝謝老師與助教這三個月的教導與協助，讓我受益匪淺，往後也期許自己持續學習，並將以前學到的專業知識融會貫通，讓職涯有更好的發展。
- 專題分工：僅製作部分專題簡報，無製作專題報告書。

5. 136 吳學賢
    - 工作崗位：目前任職於富邦銀行
    - 參與課程的動機：參與課程動機是對 AI 與半導體有顆好奇的心，希望能藉由課程建立對此領域有基礎的了解。
    - 心得分享：對此領域像是 python、機器學習、深度學習有初步概念，期待未來能藉由 AI 在工作上作進一步提升。
    - 專題分工：僅製作部分專題簡報，無製作專題報告書。

6. 098 江亮瑩
    - 工作崗位：曾任職過金融業與貿易業
    - 參與課程的動機：參與課程動機是對 Python & AI & ChatGPT &大數據分析有顆好奇的心，希望能藉由課程建立對此領域有基礎的了解，AI 產業目前正是熱門的行業，對於 AI 的技術和前景也有初步的了解未來有機會想投入與 AI 相關的領域。
    - 心得分享：過去在學校是念經濟相關學系的，每天在工作方面接觸的都是數字與人，上課內容提到的 Python、數據預測、資料視覺化分析、臉部特徵、 影像處理與 ChatGPT 內容豐富，期待未來能將過去的工作經驗結合 AI 在職涯上也多了可以發展的方向。
    - 專題分工：僅製作部分專題簡報，無製作專題報告書。

7. 178 劉威廷
    - 工作崗位：任職於超豐電子
    - 參與課程的動機：因為目前 AI 領域的火熱，希望能夠更加深入了解，提升自己在該領域的專業知識和技能，以應對未來職場需求。我期待能學習到實際應用的方法，並在工作中推動創新，提高效率。
    - 心得分享：以前並沒有學習 AI 相關領域的經驗，不過在參與本課程後，讓我對 AI 的基礎知識有了更深入的了解。課程內容豐富，涵蓋了各種技術和應用，老師講解細緻且易懂，讓我能夠快速掌握核心概念。
    - 專題分工：僅製作部分專題簡報，無製作專題報告書。

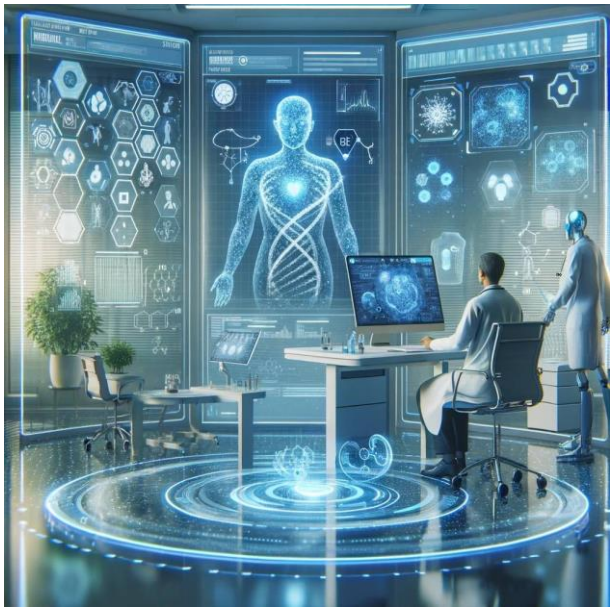
第一章　簡介

1.1 背景說明

我們的臉透過可辨識的表情瞬間傳達情感，嘴角上揚的微笑表示快樂，哀傷落淚表示悲傷等等，因此臉部訊號使我們能夠立即得知他人的感受，但在某些特殊情況下，如住院病人或有情感表達障礙的人，情緒辨識可能會受到限制。故提高情感識別能力在醫療方面有助於提升患者的治療效果，改善醫療服務品質，亦可促進個人化醫療。

在心理健康診斷與治療方面，情緒辨識技術可以用於診斷和治療心理健康問題，例如憂鬱症、焦慮症和創傷後壓力症候群，通過分析患者的表情、聲音語調及言語內容，協助醫療人員更準確地了解患者的情緒狀態，進而制定更有效的治療方案，例如：開發智能手機應用，通過每天記錄用戶的面部表情和聲音語調，持續監測其情緒變化，並提供心理健康建議或提醒用戶尋求專業幫助。另外，台灣社會即將進入超高齡化，老年護理成為發展重點之一，這極可能面臨供不應求之困境，因此配置情緒辨識功能的機器人，通過與老年人的互動，除了提供心理上的陪伴，同時也可以監測其情緒狀況，在必要時通知醫護人員或家屬，使長者得以擁有智慧家庭。

將 AI 與情緒辨識技術結合應用於醫療方面，這是當前發展的重要趨勢，因此本研究將開啟情緒辨識的黑盒子，包括資料收集、資料標記、資料輸入、機器學習、辨識輸出等等，未來也可以加入多模態情緒辨識，也就是整合語音、文字、臉部表情等資訊，透過深度神經網絡分析，達到更精準的情緒判讀。



圖一、人工智慧在醫學中的應用

1.2 痛點

1.2.1 數據隱私與安全

A. 數據敏感性：醫療和情緒數據都極為敏感，需要嚴格保護。如何確保這些數據的安全和隱私是首要挑戰。
B. 法規合規：不同地區對數據保護的法規要求不一，確保在全球範圍內合規收集和處理數據具有挑戰性。

1.2.2 技術準確性與可靠性

A. **辨識精度**：目前的情緒辨識技術在不同環境和人群中的準確性不一致，難以達到醫療應用所需的高精度。
B. **個體差異**：不同文化背景和個體之間的情緒表達差異顯著，增加了辨識的複雜性和不確定性。
C. 人類**情緒複雜**，性別、年齡、個性、生活背景、乃至不同的互動情境都會影響，但過去情緒辨識無法具體評估個體差異在其中所造成的影響，讓辨識結果不夠精確。

1.2.3 臨床驗證與用戶接受度

近來許多情緒研究顯示此通說未必正確。過去兩年，一群科學家閱讀了近一千篇關於情緒與表情的研究，得出的共同結論是：人類無法單靠臉部表情準確解讀他人的情緒，而且將情緒辨識技術應用於醫療，需要經過嚴格的臨床試驗來驗證其有效性和安全性，這一過程耗時且成本高昂。以下提供可能的解決方案：

I. **臨床試驗和研究**：進行嚴格的臨床試驗，提供數據證明情緒辨識技術的有效性和安全性，獲得醫療專業人員的信任。
II. **醫療專業人員的認可**：醫療界對新技術的接受和信任需要時間和證據的積累，特別是對於情緒辨識這類新興技術。
III. **教育和培訓**：對醫療專業人員進行系統的教育和培訓，讓他們充分了解並掌握情緒辨識技術的應用方法和優勢。
IV. **跨學科合作**：促進技術開發者與醫療專業人員之間的合作，共同優化和改進技術應用。
V. **患者知情同意**：制定透明的知情同意過程，確保患者充分了解技術的應用範圍和數據使用方式，並自願同意。

1.2.4 技術準確性

- **多模態數據融合**：結合面部表情、聲音語調、語言內容和生理信號等多種數據源，提高情緒辨識的準確性。

- **深度學習模型**：使用先進的深度學習模型，如卷積神經網絡（CNN）和長短期記憶網絡（LSTM），提升情緒辨識的效果。
- **大數據訓練**：收集大量且多樣化的訓練數據，不斷優化和更新模型，以提高辨識的精度和可靠性。

1.2.5 用戶接受度：患者是否信任並接受這種新技術

透明溝通：向患者詳細解釋情緒辨識技術的工作原理、應用目的和數據保護措施，增強他們的信任。

操作教學：優化技術的用戶界面和操作流程，提高使用便利性，特別是針對老年人和技術不熟悉的患者。

提供反饋機制：建立患者反饋機制，收集他們的意見和建議，並根據反饋進行改進。

1.3 動機

在現代醫療領域中，個人化醫療與精準醫療是提升治療效果和護理品質的關鍵方向，相較於傳統醫療方法只依賴於患者的主觀敘述和醫生的觀察，導致很多情況下帶來不確定性，而情緒為反映人類心理與生理狀態的重要指標，希望能準確辨識且即時反應，將對醫療診斷和治療上之輔助提供更為正確的方向。

1.4 目的

1.4.1 主旨

探討情緒辨識技術的方法與準確率，並透過小組討論去尋求解方， 期望深入分析情緒辨識技術的理論基礎和實際應用場景，結合臨床試驗和實證研究，為推動情緒辨識技術在醫療中的廣泛應用提供理論支持和實踐指導。同時，本研究也希望能引起更多學術界和醫療界的關注，促進相關領域的跨學科合作，共同推動醫療技術的進步與發展。

1.4.2 輸入

我們參考 kaggle 上的 Facial Recognition 資料，輸入的 dataset 為：**/kaggle/input/facial-recognition-dataset**，且輸入函數如下圖二所示，接著將設置訓練和測試數據路徑為圖三所示。

```
import numpy as np
from tqdm import tqdm #用來顯示進度條以及展示每一輪（iteration）所耗費的時間
import cv2
import os #操作系統相關的操作，如文件和目錄管理
import tensorflow as tf
import tensorflow_hub as hub
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelBinarizer #標籤二值化
from sklearn.model_selection import train_test_split #拆分數據集
from sklearn.metrics import accuracy_score, confusion_matrix #評估模型性能
import itertools #用於迭代器生成和操作
import plotly.graph_objs as go #用於創建圖形對象
from plotly.offline import init_notebook_mode, iplot #在筆記本中顯示圖表
from plotly import tools #用於輔助圖表生成
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D #從Keras庫中導入不同的層類型，用於構建神經網絡
from keras.preprocessing.image import ImageDataGenerator #圖像數據增強
from keras.applications.vgg16 import VGG16, preprocess_input #圖像預處理
from keras import layers #構建神經網絡層
from keras.models import Model, Sequential #構建和訓練神經網絡模型
from keras.optimizers import Adam, RMSprop
from keras.callbacks import EarlyStopping #提前停止訓練以防止過擬合
from keras.preprocessing.image import ImageDataGenerator
init_notebook_mode(connected=True) #在筆記本中顯示圖表的功能
RANDOM_SEED = 123 #設置隨機種子，以確保結果的可重現性
```

圖二、本專案運用之函數等資料

```
TRAIN_DIR = ('../input/facial-recognition-dataset/Training/Training/')
TEST_DIR = ('../input/facial-recognition-dataset/Testing/Testing/')
```

圖三、本案所設置之數據路徑

1.4.3 模型功能

本專案**所使用的模型為 VGG16**，其主要特點是網路結構比較深，且捲積層和池化層的數量都比較多，使得網路可以學習到更加高層次的抽象特徵。 此外，VGG16 的捲積層都採用 3x3 的卷積核，這樣可以保證在不增加計算量的情況下，增加了網路的深度和寬度，提高了特徵提取的效率和準確性。從函數 'keras.applications.vgg16' 導入模型 VGG16，該模型架構是卷積+卷積+卷積+池化，**第二章節中「模型介紹—2.3.2 建立基礎模型」將有具體詳細介紹**。模型功能如下：

(1) **減少參數的數量**，可加快計算速度。
(2) 參數個數減少可以**防止過擬合**。
(3) 網絡層次深，由於每個卷積都會有一個激活函數，故**可擬合更複雜的數據**。
(4) **Block 結構**：每一個 Block 中含有幾個捲積層和一個池化層，圖四清楚可見由 Block 所劃分出的 VGG16 之結構圖。

圖四、 Block 劃分的 VGG16 的結構圖

(5) **採用小卷積核**代替大卷積核：使用多個小卷積核 3x3 來代替 其他大型卷積核，例如：AlexNet(5x5)，如此可達到相同的感受也同時減少參數量，在不考慮 bias 的情況下，3x3 的卷積層只需要 3x3x2=18 個參數量 。

1.4.4 輸出

本專題使用函數 'plot_confusion_matrix '，繪製**混淆矩陣**，用於評估分類模型的性能，在 'model.compile()' 中，設置了模型的訓練配置，**詳細可參考圖十二**，包括：

- 損失函數：categorical_crossentropy
- 優化器：選擇了 Adam 優化器，設置學習率為 1e-4
- 評估指標：metrics=['accuracy']
- history = model.fit()' 中，使用 fit 方法開始訓練模型
- 最後一行：'history'將會儲存訓練過程中的損失和準確率等指標

測試集上驗證模型，並計算模型的準確率、混淆矩陣等指標，本專題導入 f1_score、precision_score 和 recall_score 函數，用於評估分類模型的性能指標。根據 **Loss 下降的程度與 Accuracy 逐漸上升**得知，training data 是有效被學習，但 Testing accuracy 的準確率卻只有 0.61，推測其可能有 overfitting，詳細輸出結果請參考第三章圖十三與圖十四。

1.4.5 可達成的目的(預期效益)

I. **高準確度的情緒分類**：本專題所使用 VGG16，此為一種深層卷積神經網路（CNN），其多層結構和足夠的參數量等特性，使其擁有強大的特徵提取能力，從而在情緒分類任務中獲取較高準確率，關於 VGG16 之模型介紹詳閱第二章之第三節。

II. **可解釋性和視覺化**：VGG16 的結構相對較為簡單且好理解，因其特徵圖的可視化和解釋性相對較好，在本專題幫助組員們理解模型的運作原理以及展現情緒辨識中的圖像特徵提取方式等等。

III. **應用場景廣泛**：情緒辨識可以應用於多個領域，如人機交互、心理健康監控、智能客服、教育等。通過準確的情緒辨識，可以提升這些應用的智能化和人性化程度，可見第四章第三節之未來展望。

IV. **實現自動化情緒分析**：本專題希望在未來研究加入多模態情緒分析，一來可以大幅提升系統性能和可靠性，二來更全面地捕捉和理解情緒。

第二章　系統架構與技術

2.1 數據資料說明

2.1.1 來源：本次使用的測試集資料主要來自 Kaggle 的 Facial Recognition Dataset 的測試集資料，其餘則是網路上搜尋的圖片。使用情緒圖片計有 Angry 等六類 7067 張圖片，另外加上網路圖片各類型 100 張，合計 7667 張圖片。

2.2 資料型態與前處理

2.2.1 資料型態

資料來源：利用 Vgg16 卷積神經網路技術提取臉部特徵，並利用這些特徵分辨出圖片中的人物表情是 Angry、Fear、Happy、Neutral、Sad 與 Surprise。

2.2.2 資料前處理

指的是在進行機器學習或深度學習之前，對原始的資料進行處理或轉換的過程，為了確保訓練模型時使用的資料品質、一致性與合適程度以提高模型性能。資料清理與資料型態調整，需被處理的資料指的就是無法透過數學運算的，可以簡單分成幾種類型：

- ✧ 異常值：刪除、視為缺失值，用處理缺失值的方法進行處理。
- ✧ 缺失值：補上固定值 0 或直接刪除有缺失值的資料樣本。
- ✧ 類別化：獨熱編碼 (one-hot encoding) 就是解決這的問題的方法。
- ✧ 正規化：將資料縮放到相同的數值範圍，如 0 到 1 之間。

2.2.3 圖像數據加載和預處理

首先定義一個函數來**加載數據**，參數為數據路徑和圖像大小，使用兩個空列表分別為：X 存儲圖像數據，y 存儲圖像標籤，設置一個計數器 i 和一個字典 labels，labels = dict()：用於存儲類別標籤，**圖像數據加載和預處理**如圖五所示：

```
def load_data(dir_path, IMG_SIZE):

    X = []
    y = []
    i = 0
    labels = dict()
    for path in tqdm(sorted(os.listdir(dir_path))):
        if not path.startswith('.'):   #排除隱藏文件和目錄
            labels[i] = path  #將類別標籤對應的目錄名稱存儲在字典 labels 中
            for file in os.listdir(dir_path + path):  #遍歷每個子目錄中的文件
                if not file.startswith('.'):  #排除隱藏文件
                    img = cv2.imread(dir_path + path + '/' + file)  #使用 OpenCV 讀取圖像文件
                    img = img.astype('float32') / 255  #將圖像數據轉換為浮點數並標準化到 [0, 1] 範圍
                    resized = cv2.resize(img, IMG_SIZE, interpolation = cv2.INTER_AREA)  #調整圖像大小為指定的 IMG_SIZE
                    X.append(resized)  #將調整大小的圖像添加到列表 X 中
                    y.append(i)  #將類別標籤添加到列表 y 中
            i += 1  #跑完一個類別後，遞增計數器 i。
```

圖五、圖像數據加載和預處理

2.2.4 繪製圖像

此步驟為**資料前處理中的數據可視覺化**，如「①初始化和設置：定義函數和初始化參數；②遍歷類別：所有類別的索引，選取每個類別的前 n 張圖像；③設置圖表：計算行數和列數，創建新圖表並設置大小；④繪製 subplot：創建子圖，顯示圖像並隱藏刻度；⑤設置標題和顯示圖表：設置圖表標題為類別名稱並顯示圖表」，如圖六所示。使用"plot_samples"函數來繪製樣本圖像，在此設定圖像數量為 5 的倍數，亦可自行更改其數量，如圖七。

```python
def plot_samples(X, y, labels_dict, n=50):

    for index in range(len(labels_dict)):
        imgs = X[np.argwhere(y == index)][:n]  #找出所有屬於當前類別的圖像，並取前 n 張
        j = 5  #設定每行顯示的圖像數量為 5
        i = int(n/j)  #計算需要的行數

        plt.figure(figsize=(10,3))
        c = 1
        for img in imgs:
            plt.subplot(i,j,c)
            plt.imshow(img[0])

            plt.xticks([])
            plt.yticks([])
            c += 1
        plt.suptitle(labels_dict[index])  #設置整個圖表的標題為當前類別的名稱
        plt.show()
```

圖六、繪製圖像樣本

```python
plot_samples(X_train, y_train, train_labels, 5)
```

圖七、輸出樣本圖像

2.2.5 訓練數據標籤轉換

在機器學習和深度學習中，將**類別標籤編碼為二進制格式**是常見的步驟。to_categorical() 方法可以將一個包含整數標籤的 NumPy 數組或向量轉換為二進

制格式的矩陣，其中每一列對應一個類別。這樣可以更方便地將標籤輸入到神經網絡中，本專案使用 Keras 中的 'to_categorical' 函數，該函數用於**將類別標籤進行獨熱編碼 (One-Hot Encoding)**，如圖八所示：

```
# 'to_categorical' 函數，該函數用於將類別標籤進行獨熱編碼 (One-Hot Encoding)
from keras.utils.np_utils import to_categorical

Y_train = to_categorical(y_train, num_classes=6) # 有6個類別，故 num_classes 設置為 6。
Y_train.shape # 獨熱編碼的標籤 Y_train 的形狀
```

```
(28273, 6)
```

```
Y_test = to_categorical(y_test, num_classes=6)
Y_test.shape
```

```
(7067, 6)
```

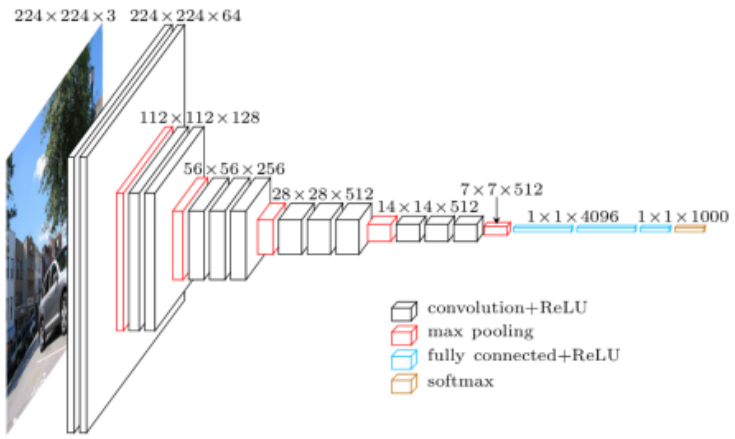
圖八、將標籤轉換為 One-Hot Encoding

2.3 模型介紹：VGG16 model

2.3.1 模型設計

本次所使用的 CNN 模型為——VGG16，是由牛津大學的 Visual Geometry Group 提出的一種深度卷積神經網絡模型，該模型在 2014 年的 ImageNet 挑戰賽中表現優異。**VGG16 的結構非常有規則且簡潔**，主要由以下幾個部分組成：

- **輸入層（Input Layer）**：接受形狀為 (224, 224, 3) 的彩色圖像，這意味著圖像的寬和高都是 224 像素，有 3 個顏色通道（紅、綠、藍）。
- **卷積層（Convolutional Layers）**：VGG16 由 13 個卷積層組成，每個卷積層使用 3x3 的小卷積核，並且 stride 為 1，這使得輸出與輸入的空間尺寸保持一致。 卷積層之間使用 ReLU 激活函數來增加網絡的非線性。
- **池化層（Pooling Layers）**：在每個卷積塊之後，都有一個 2x2 的最大池化層，這些池化層負責減小空間維度，同時保留重要特徵。
- **全連接層（Fully Connected Layers）**：在卷積層和池化層之後，VGG16 包含三個全連接層，其中前兩個全連接層每個都有 4096 個神經元，第三個全連接層則有 1000 個神經元。不過此次分類只需分 6 類，所以沒有用到此模型內建的全連接層。



圖九、VGG16 模型示意圖

2.3.2 建立基礎模型

圖十顯示我們建立了一個基於 VGG16 架構的模型 base_model，**用於圖像處理的特徵提取**，VGG16 是一種深度卷積神經網路，以其在圖像分類聞名，這樣的模型可以在後續的圖像分類、物體檢測或者其他視覺任務中作為基礎模型進行進一步的訓練或應用。

```python
from keras.applications.vgg16 import VGG16

base_model = VGG16(
        weights=None,  #表示不使用預訓練的權重，而是隨機初始化權重
        include_top=False,   #自定義的特徵提取器
        input_shape=IMG_SIZE + (3,)  #表示輸入圖像是彩色 RGB
    )

base_model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 48, 48, 3)]       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 48, 48, 64)        1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 48, 48, 64)        36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 24, 24, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 24, 24, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 24, 24, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 12, 12, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 12, 12, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 6, 6, 256)         0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 6, 6, 512)         1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 3, 3, 512)         0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 1, 1, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
```

圖十、VGG16 架構建立模型

2.3.3 建立完整的深度學習模型

這段程式碼組合了卷積神經網路的特徵提取部分（使用 VGG16 的卷積基底）、全連接層來進行更高層次的特徵學習，並**最終通過 softmax 分類器來實現多類別分類任務**，如圖時所示加入 Flatten、全連接層(Dense)：在本專案中添加一個具有 1000 個神經元的全連接層，使用 ReLU 激活函數、加入 Dropout 是

為了防止過度擬合，以及最後一層全連接層(Dense)，並且使用 softmax 激活函數進行多類別分類等等，以上敘述如圖十一可見。

```
NUM_CLASSES = 6          # 6種不同的情緒樣本

model = Sequential()      # 使用 Sequential 模型將基礎模型（base_model）作為第一層。
model.add(base_model)     # 將 VGG16 基模型添加到序冒模型中
model.add(Flatten())      # 將多維數據轉為一維
model.add(Dense(1000, activation='relu'))     # 添加了一個具有 1000 個單元的全連接層（Dense），並使用 ReLU 激活函數。
model.add(Dropout(0.4))           # 在訓練過程中以 40% 的概率隨機丟棄一些神經元，防止過擬合。
model.add(Dense(NUM_CLASSES, activation='softmax'))     # 添加具有 softmax 激活的輸出層
```

圖十一、深度學習模型

2.3.4 模型參數調校及優化

優點：

- ❖ 被視為至今為止的優秀視覺模型之一。
- ❖ 擁有大量超參數的模型。
- ❖ 最重要的概念就是大量使用 3X3 的 Conv layers、較小的 stride(strides=1) 以及 Pooling (2X2)。

缺點：

- ❖ 在 Colab 上跑 VGG 不開 GPU train 速度超級慢。
- ❖ 架構相對較深且具有大量的參數 ➔ 需要更多的**計算資源和時間**。
- ❖ 處理大型圖像數據集時可能會遇到**過度擬合（overfitting）**。
- ❖ 不太適合於嵌入式設備或需要低計算資源的應用場景。

跟別人比較優劣勢：

➔ 可能是換成用 **Resnet** 測試或是利用**多模態**提高準確率

◆ **Resnet**

- ❖ 具 residual 結構，並搭建超深的網路結構（突破 1000 層）
- ❖ 使用 Batch Normalization 加速訓練（丟棄 dropout）

◆ **多模態**

- ❖ 擁有二個以上的模態：輸入人臉表情及文字含義，例如：圖像以 CNN 為主，文字以 RNN、 GPT 為主，而輸出就是一串數字向量。
- ❖ 同時有人的臉部表情與說話的文字內容。
- ❖ Feature fusion：此階段為**整合人臉及文字特徵**，兩種做法：分別是 Model free 以及 Model based，目前後者較常見。

第三章　效能評估

3.1 環境與運行時間(說明執行環境、效能評估條件總結)

3.1.1 此處定義了一個函數'deep_model'，該函數接受五個參數，分別為：

- model：深度學習模型，通常是一個 Keras 模型的實例。
- X_train：訓練數據的特徵，是一個 numpy 數組。
- Y_train：訓練數據的標籤，是獨熱編碼格式（numpy 數組）。
- epochs：訓練的迭代次數，此處設定為 40。
- batch_size：每個批次的樣本數量，此處設定為 64。

另外，在'model.compile()'中，設置模型的訓練配置，包括內容如下，亦可參考圖十二，該圖為模型依照 training data 得到之準確度。

- 損失函數：categorical_crossentropy
- 優化器：選擇了 Adam 優化器，設置學習率為 1e-4
- 評估指標：metrics=['accuracy']。
- 'history = model.fit()'中，使用 fit 方法開始訓練模型。
- 最後一行：'history'將會儲存訓練過程中的損失和準確率等指標。

3.1.2 執行環境

- 運行時間：1 epochs 所需要的平均運行時間為 446 秒，總共所需時間為 18066 秒，相當於 5.18 小時。
- 工作環境：使用 Jupyter Notebook，Python 3.8 / Keras 2.6.0/ Tensorflow2.6.0。

```python
def deep_model(model, X_train, Y_train, epochs, batch_size):

    model.compile(
    loss='categorical_crossentropy',
    optimizer=Adam(learning_rate=1e-4),
    metrics=['accuracy'])

    history = model.fit(X_train
                        , Y_train
                        , epochs=epochs
                        , batch_size=batch_size
                        , verbose=1)
    return history
```

```python
epochs = 40
batch_size = 64

history = deep_model(model, X_train, Y_train, epochs, batch_size)
```

```
442/442 [==============================] - 447s 1s/step - loss: 0.0566 - accuracy: 0.9820
Epoch 32/40
442/442 [==============================] - 444s 1s/step - loss: 0.0568 - accuracy: 0.9813
Epoch 33/40
442/442 [==============================] - 445s 1s/step - loss: 0.0519 - accuracy: 0.9834
Epoch 34/40
442/442 [==============================] - 446s 1s/step - loss: 0.0480 - accuracy: 0.9837
Epoch 35/40
442/442 [==============================] - 446s 1s/step - loss: 0.0465 - accuracy: 0.9841
Epoch 36/40
442/442 [==============================] - 446s 1s/step - loss: 0.0511 - accuracy: 0.9827
Epoch 37/40
442/442 [==============================] - 447s 1s/step - loss: 0.0403 - accuracy: 0.9859
Epoch 38/40
442/442 [==============================] - 447s 1s/step - loss: 0.0454 - accuracy: 0.9847
Epoch 39/40
442/442 [==============================] - 447s 1s/step - loss: 0.0413 - accuracy: 0.9867
Epoch 40/40
442/442 [==============================] - 445s 1s/step - loss: 0.0407 - accuracy: 0.9862
```

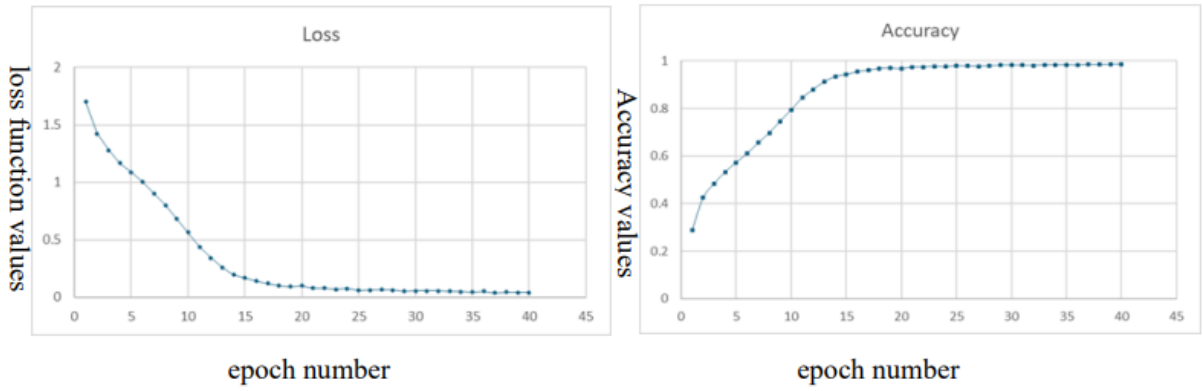
圖十二、設置訓練模型

3.2 效能評估條件總結

圖十三分別顯示了一個學習模型在訓練過程中的損失值(Loss)和準確率(Accuracy)隨著訓練輪數(epoch)的變化。

左圖為 loss function 與迭代次數的關係，在初始幾個訓練輪數中，損失值快速下降，**表明模型正在學習並改進其預測**，在大約 20 個訓練輪數之後，損失值趨於穩定，接近 0.5 左右，這表明模型的性能達到了一個穩定狀態，不再有顯著的改進。其下降代表:**training data 有良好的學習效果**。

右圖為 Accuracy 與迭代次數的關係，在初始幾個訓練輪數中，準確率快速上升，表明此模型**正在學習如何更好的分類**。在大約 10 個訓練輪數之後，準確率趨於平穩，可見其值接近 1.0，這表明模型已達到較高的準確率，不再有顯著改變。而準確度高意味著訓練此 model 可以**準確的辨識 training data 中的圖片**。



圖十三、loss 和 accuracy 表現結果

3.3 效能衡量指標

3.3.1 說明正確率（Accuracy）

圖十四為使用 testing data 來驗證模型之成效，從驗證結果可見使用 testing data 得到的 testing accuracy 僅有 0.61，推測可能的原因是因 VGG16 的參數量大，可能將 **trainging data 內的某特徵值考慮進去導致判斷 testing data 時無法辨別，最終導致過擬合**。有可能只藉由影像辨識情緒辨識的準確度不夠高，需要搭配文字及聲音的多模態將模型準確率提高。

```
# Validate on test set
from sklearn.metrics import f1_score, precision_score,  recall_score
from sklearn.metrics import classification_report

predictions = model.predict(X_test)
y_pred = [np.argmax(probas) for probas in predictions]


accuracy = accuracy_score(y_test, y_pred)
print('Test Accuracy = %.2f' % accuracy)

confusion_mtx = confusion_matrix(y_test, y_pred)
cm = plot_confusion_matrix(confusion_mtx, classes = list(test_labels.items())
```

```
Test Accuracy = 0.61
```

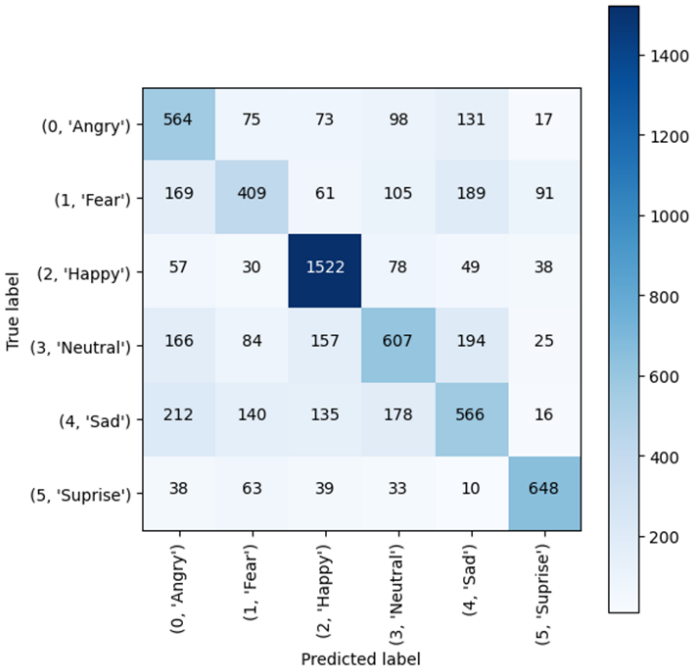
圖十四、Testing Accuracy

3.3.2 混淆矩陣（Confusion Matrix）

圖十五為此**模型的混淆矩陣**，其顯示了模型在六種類別（Angry、Fear、Happy、Netutral、Sad、Surprise）上的**實際標籤和預測標籤之對比結果**。矩陣中的數字表示對應位置上實例的數量。例如：位於(2,2)的位置表示實際標籤為 Happy，且被模型正確預測為 Happy 的實例數為 1522。

混淆矩陣總結，**模型在 Happy 和 Surprise 兩類的情感上有較好表現**，而在 **Angry 和 Fear 情感類別上，模型的錯誤預測較多**，因其會將 Angry 誤認為是 Sad，也將 Fear 預測失誤成 Sad。以下分別顯示各類情感的混淆矩陣結果：

- **Angry**
  - 正確預測：有 564 個。
  - 錯誤預測；有 75 個預測為 Fear、73 個預測為 Happy，98 個預測為 Neutral，131 個預測為 Sad，17 個預測為 Surprise。
- **Fear**
  - 正確預測：有 409 個。
  - 錯誤預測：有 169 個預測為 Angry、61 個預測為 Happy，105 個預測為 Neutral，89 個預測為 Sad，91 個預測為 Surprise。
- **Happy**
  - 正確預測：有 1522 個。
  - 錯誤預測：有 57 個預測為 Angry、30 個預測為 Fear，78 個預測為 Neutral，49 個預測為 Sad，38 個預測為 Surprise。
- **Sad**
  - 正確預測：有 566 個。
  - 錯誤預測：有 212 個預測為 Angry、140 個預測為 Fear，135 個預測為 Happy，178 個預測為 Neutral，164 個預測為 Surprise。
- **Surprise**
  - 正確預測：有 648 個。
  - 錯誤預測：有 38 個預測為 Angry、63 個預測為 Fear，39 個預測為 Happy，33 個預測為 Neutral，10 個預測 Sad。



圖十五、Confusion matrix

3.3.3 說明準確率、回召率與 F1 score

最後將此模型做**性能綜合評估**，包括準確率 (precision)：表示被預測某一類別的樣本實際屬於該類別的比例；召回率 (recall)：表示實際屬於某一類別的樣本被正確預測該類別的比例；F1-score：準確率和召回率的調整平均數，平衡兩者的綜合指標，其結果如圖十六所示。

將圖十六轉為表格形式呈現，且依 F1-score 分數高低排序有利於結果討論，因此發現在該模型中，Happy 與 Surprise 被辨別率最好，其 F1-score 分別為 0.81 和 0.78。Angry 與 Neutral 被辨別率介於中間，其 F1-score 均為 0.52。Sad 與 Fear 被辨別率最差，其 F1-score 分別為 0.45 和 0.48，如表一所示：

	precision	recall	f1-score
Angry	0.47	0.59	0.52
Fear	0.51	0.40	0.45
Happy	0.77	0.86	0.81
Neutral	0.55	0.49	0.52
Sad	0.50	0.45	0.47
Suprise	0.78	0.78	0.78

圖十六、各類情緒之準確率、回召率與 F1 score

	Happy	Surprise	Angry	Neutral	Sad	Fear
Precision	0.77	0.78	0.47	0.55	0.50	0.51
Recall	0.86	0.78	0.59	0.49	0.45	0.40
F1-score	0.81	0.78	0.52	0.52	0.47	0.45

表一、各類情緒之 F1-score 比較

第四章　結論

4.1 反思

- VGG-16模型複雜性：參數量大，計算資源需求高，限制其在資源有限環境中的應用。
- 模型優化需求：面對複雜圖像分類任務，VGG-16可能不足，提示我們需要模型創新或結合其他神經網路。

4.2 創新

- VGG-16 的 3x3 卷積特徵：小卷積核創新應用，保持小同時增加特徵抽取深度，有效理解圖像的複雜特徵。
- 重疊最大池化技術：引入重疊池化，提升性能，確保特徵的穩健性，減少參數量的同時增強模型效果。

4.3 未來展望(工作)

- 優化 VGG-16 模型結構：研究減少計算需求的方法，維持或提升模型預測準確性，以實現更高效能。
- 擴展應用領域：將模型應用於影片處理、醫學影像分析，以驗證其在不同任務中的適應性和效果。
- 深度學習與現實結合：探索將 VGG-16 應用於自動駕駛、智能監控等複雜場景，解決實際問題，推動技術創新。
- 機器學習研究啟示：為學者和工程師提供關鍵反思，指出模型改進方向及未來研究的潛在價值。

第五章　參考資料

1. Redação Jornal de Brasília (Feb 15, 2024). Como a IA interfere na área de medicina. https://jornaldebrasilia.com.br/noticias/opiniao/como-a-ia-interfere-na-area-de-medicina/

2. 個人化醫療：客製化醫療解決方案的 6 個重要面向
   個人化醫療：客製化醫療解決方案的 6 個重要面向 (julienflorkin.com)

3. CNN 學習筆記-VGG 架構.May 25, 2020.Andy_Chen
   https://medium.com/@x22413128/cnn%E5%AD%B8%E7%BF%92%E7%AD%86%E8%A8%98-vgg%E6%9E%B6%E6%A7%8B-f055ab3ec803

4. VGG16 學習筆記 (2018-07-26) 韓鼎の個人網站
   https://deanhan.com/2018/07/26/vgg16/?source=post_page-----f055ab3ec803----------

5. ResNet——CNN 經典網路模型詳解（pytorch 實現）2020-04-29 浩波的筆記
   https://blog.csdn.net/weixin_44023658/article/details/105843701

6. 人類的情緒變化，AI 真的能辨識出來，還能分析應用？2023/08/12 AIF EVENT
   https://edge.aif.tw/aicafe-0628-emotion-recognition/